Week\_6\_Homework

Olivia Townsend

9/27/2017

# Create a function to download tide data from the NOAA website with a pre-selected tide station and date range

# coops\_tide\_ht\_retrieval.R  
# Author: Luke Miller Feb 2011  
# Updated 2017-09-26 to handle newer ERDDAP format  
################################################################################  
# Run this script as-is. It will prompt you to enter the appropriate  
# station and date values when needed. You can only request less than 1 month of   
# data at a time, so if you get an error returned, try asking for slightly less  
# data.  
  
# This script will download a set of verified tide height data from a NOAA   
# CO-OPS DODS/OPeNDAP server and parse it into a data frame to be saved to disk   
#  
# Station list: http://opendap.co-ops.nos.noaa.gov/stations/index.jsp  
# Scroll through that list to find your station ID.   
  
# OPeNDAP server gateway: https://opendap.co-ops.nos.noaa.gov/erddap/index.html  
# The gateway has links to several other data types available including   
# water temperature, air temperature, wind etc.   
# See http://docs.opendap.org/index.php/UserGuideOPeNDAPMessages for info on  
# structuring OPeNDAP queries.  
#  
# Six-minute water level data can only be retrieved 1 month at a time. Other  
# fields such as water temperature could return up to 1 year at a time. This   
# script only deals with 6-minute tide height data, but could serve as a guide   
# for accessing other data types from NOAA CO-OPS.   
# The tide height is reported in meters and time in UTC (Greenwich).  
  
###############################################################################  
library(RCurl)

## Loading required package: bitops

#noquote(print("Enter NOAA tide station number (i.e. Monterey = 9413450): "))  
#station = scan("",what = character(),nlines = 1)  
#noquote(print("Enter start date (format: 20080123 = Jan 23, 2008): "))  
#startdate = scan("",what = character(),nlines = 1) #get one line of values  
#noquote(print("Enter end date (format: 20081231 = Dec 31, 2008): "))  
#enddate = scan("",what = character(),nlines = 1)  
  
#OPeNDAP query for 6-minute verified water level looks like this (on 1 line):  
# https://opendap.co-ops.nos.noaa.gov/erddap/tabledap/  
# IOOS\_SixMin\_Verified\_Water\_Level.asc?STATION\_ID  
# %2CDATUM%2CBEGIN\_DATE%2CEND\_DATE%2Ctime%2CWL\_VALUE%2CSIGMA%2CI%2CF%2CR%2CT&  
# STATION\_ID=%229413450%22&  
# DATUM=%22MLLW%22&  
# BEGIN\_DATE=%2220080123%22&  
# END\_DATE=%2220080130%22  
#####################################################  
######## DON'T CHANGE ANYTHING BELOW THIS LINE ####################  
#The parts of the url to be assembled  
  
NOAAfunction<-function(station,startDate,endDate){  
 url1 = "https://opendap.co-ops.nos.noaa.gov/erddap/tabledap/"  
 url2 = "IOOS\_SixMin\_Verified\_Water\_Level.asc?"  
 url3 = "STATION\_ID%2C" #return stationId  
 url4 = "DATUM%2C" #return datum  
 url5 = "time%2C" #return record date-time  
 url6 = "WL\_VALUE%2C" #return water level value  
 url7 = "I%2C" #return quality flag  
 url8 = "F%2C" #return quality flag  
 url9 = "R%2C" #return quality flag  
 url10 = "T" #return quality flag  
 #The remaining parts of the url specify how to filter the data on the server   
 #to only retrieve the desired station and date range. Values must be enclosed  
 #in ascii double-quotes, which are represented by the code %22  
 # Do not change any values here, do not insert your own values here.  
 url11 = "&STATION\_ID=%22" # station ID goes here  
 url12 = "%22"  
 url13 = "&DATUM=%22MLLW%22" # we want MLLW as the datum  
 url14 = "&BEGIN\_DATE=%22" # start date gets put in here  
 url15 = "%22"  
 url16 = "&END\_DATE=%22" # end date gets put in here  
 url17 = "%22"  
 ####### DON'T CHANGE ANYTHING ABOVE THIS LINE ############  
 ##############################################  
   
 #Assemble the URL  
 urltotal = paste(url1,url2,url3,url4,url5,url6,url7,url8,url9,url10,url11,  
 station,url12,url13,url14,startDate,url15,url16,endDate,url17,sep ="")  
   
 #Download the data  
 cat("Contacting server...\n"); flush.console()  
 dat = getURL(urltotal) #use RCurl to retrieve text into a vector 'dat'  
 cat("Data returned...\n"); flush.console()  
 Sys.sleep(2) #If you access data in a loop, be courteous and give the server  
 #a short break between requests  
 #cleanup  
 rm(url1,url2,url3,url4,url5,url6,url7,url8,url9,url10,url11,url12,url13,url14)  
 rm(url15,url16,url17)  
   
 con = textConnection(dat) #create text Connection to dat vector  
 all.lines = readLines(con) #read lines of text into separate slots in a vector  
 close(con) #close connection to dat vector  
  
if (length(grep('^Error',all.lines))>0) { #check for error in retrieval  
 cat("There was an error...\n")  
 cat(dat,"\n") #print contents of dat to show error  
 flush.console()  
} else { #retrieval was successful, parse the text  
   
 #The column headers are typically preceded by a line of dashes  
 headerlines = grep("^--------",all.lines) #find index of headers (-1)  
   
 #read column header names into a vector  
 con = textConnection(dat)  
 headers = scan(con, skip = headerlines, nlines = 1, sep = ",",  
 what = "character", strip.white = TRUE)  
 close(con)  
   
 #read rest of the data into a data frame 'df'  
 con = textConnection(dat)  
 df = read.table(con, skip = headerlines+1, sep = ",", header = FALSE,  
 quote = "\"", col.names = headers, strip.white = TRUE,  
 stringsAsFactors = FALSE)  
 close(con)  
   
 ###########################################################################  
 #The following operations will need to be altered if you change the   
 #fields or data type being returned by the OPeNDAP server  
 # The time column should be a numeric value, representing elapsed   
 # seconds since midnight, Jan 1, 1970 in the GMT (UTC) time zone.   
 df[,3] = as.POSIXct(df[,3], tz = 'GMT', origin = '1970-1-1 00:00')  
   
 #Give the columns shorter names  
 names(df) = c("stationId","datum","TimeUTC","TideHT","Flag.Inferred",  
 "Flag.Flat.Tol","Flag.Rate.Tol","Flag.Temp.Tol")  
   
 #Uncomment this if you want to plot the data  
# plot(df$TimeUTC, df$TideHT, type = "l",  
# xlab = "Date",ylab = "Tide Height, meters")  
   
 #Save data automatically to a .csv file.   
 #filename = paste("Station\_",station,"\_",startdate,"-",enddate,  
 # ".csv",sep = "") #make file name  
 #write.csv(df,filename,row.names = FALSE, quote = FALSE) #write file to disk  
 #cat("Saved to ",getwd(),"/",filename,"\n",sep = "")  
 #flush.console()  
   
 #Alternate file save method lets user specify file name at run time  
 #Uncomment this if you wish to use it instead of the automated file   
 #output above  
# write.csv(df,file.choose(),row.names = FALSE, quote = FALSE)  
   
 #cleanup  
 #rm(dat,con,all.lines,startdate,enddate,filename,headerlines, headers)  
   
 } #end of if-else statement  
invisible(df)  
  
} #end of function definition NOAAfunction

# Create a plotting function for the tide data

The function includes appropriate axis labels, gridlines, and formatted title (month spelled out and 4-digit year).

plotTides<-function(df){  
 date<-df$TimeUTC[1]  
 date<-as.Date(date)  
 date<-format(date,format="%B %Y")  
 plot(x = df$TimeUTC,y = df$TideHT,type = 'l',xlab = "Date (month, day)",ylab = "Tide Height (m)",col="blue",main = date)  
 grid(nx=NA,ny=NULL,col = "darkgray")  
}

# Let's plot tide data for August, September, and October 2016

## First import the data using NOAAfunction

#assign data frame for August 2016, call it aug  
aug=NOAAfunction(9413450,20160801,20160831)

## Contacting server...  
## Data returned...

#sept for September 2016  
sept=NOAAfunction(9413450,20160901,20160930)

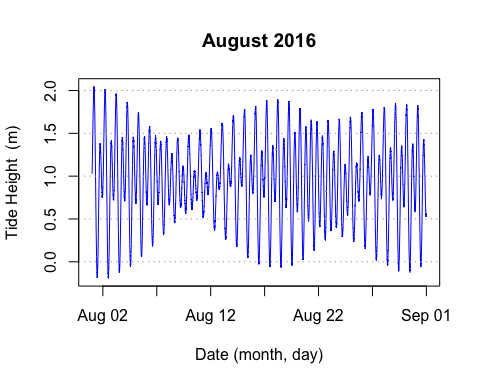
## Contacting server...  
## Data returned...

#oct for October 2016  
oct=NOAAfunction(9413450,20161001,20161031)

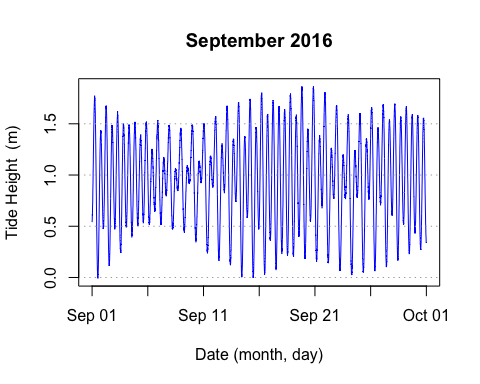
## Contacting server...  
## Data returned...

## Then create the plots using plotTides

plotTides(aug)



plotTides(sept)



plotTides(oct)

